|  |
| --- |
|  |
|  |

**Week 1: Git Basics**

**Day 1: Introduction to Git**

* **What is Git?**
  + A distributed version control system for tracking changes and enabling collaboration.
* **Why Use Git?**
  + Tracks every change to your codebase.
  + Allows for branching and merging.
  + Facilitates collaboration among developers.
* **Installing Git**
  + Download for [Windows/Mac/Linux](https://git-scm.com/downloads).
  + Set up your identity:
  + git config --global user.name "Your Name"
  + git config --global user.email "your.email@example.com"
* **Git Workflow Overview**
  + **Working Directory**: Local files.
  + **Staging Area**: Prepare files for commit.
  + **Repository**: Stores committed changes.

**Day 2: Basic Commands**

* Initialize a new repository:
* git init
* Add files to the staging area:
* git add <file> # Add specific file
* git add . # Add all files
* Commit changes:
* git commit -m "Commit message"
* Check status and history:
* git status
* git log

**Day 3: Branching and Merging**

* **What Are Branches?**
  + Enable isolated feature development.
* **Key Commands:**
  + Create a branch:
  + git branch feature-branch
  + Switch to a branch:
  + git switch feature-branch
  + Merge a branch into the main branch:
  + git checkout main
  + git merge feature-branch

**Day 4: Undoing Changes**

* Revert changes in a file:
* git restore <file>
* Unstage a file:
* git restore --staged <file>
* Reset to a previous commit:
* git reset --hard <commit\_hash>

**Week 2: GitHub Basics**

**Day 1: Introduction to GitHub**

* **What is GitHub?**
  + A platform for hosting Git repositories with collaboration tools like pull requests.
* **Creating a Repository on GitHub**
  + Steps:
    - Click **New Repository**.
    - Enter name, description, and visibility.
    - Click **Create Repository**.

**Day 2: Connecting a Local Repository**

* Add a remote repository:
* git remote add origin https://github.com/username/repository.git
* Push changes:
* git push -u origin main

**Day 3: Collaboration**

* **Forking**
  + Creates a copy of a repository in your GitHub account.
  + Clone your fork locally:
  + git clone https://github.com/username/forked-repo.git
* **Pull Requests**
  + Push changes to your fork and submit a pull request to propose changes to the original repository.

**Day 4: Conflict Resolution**

* **What Are Merge Conflicts?**
  + Occur when multiple branches modify the same section of a file.
* **Resolving Conflicts**
  + Identify conflicts during a merge:
  + git merge branch-to-merge
  + Resolve conflicts manually and mark them as resolved:
  + <<<<<<< HEAD
  + Your changes
  + =======
  + Their changes
  + >>>>>>> branch-to-merge
  + Add and commit resolved files:
  + git add file.txt
  + git commit -m "Resolve merge conflict"

**Day 5: Advanced Features**

* **GitHub Issues**
  + Track bugs, features, and tasks.
  + Categorize issues using labels and assign tasks to team members.
* **GitHub Actions**
  + Automate workflows like CI/CD.
  + Example CI workflow:
  + name: CI
  + on: [push]
  + jobs:
  + build:
  + runs-on: ubuntu-latest
  + steps:
  + - uses: actions/checkout@v3
  + - run: npm test

**Day 6: Practice Project**

1. **Set Up a Local Project**
   * Create and initialize a project locally with Git.
   * Push it to GitHub.
2. **Collaboration Exercise**
   * Add collaborators.
   * Practice pull requests and conflict resolution.
3. **Deploy the Project**
   * Use GitHub Pages to host your project.

This outline provides a clear, structured presentation of Git and GitHub concepts with practical examples and hands-on activities. Let me know if you need further refinements or visual elements!